MEADOW: Mixture of Experts for Abundant DOmains on WILDS iWildCam

CS 8803 DML Final Project Report

Richard So
School of Interactive Computing
Georgia Institute of Technology
rso31@gatech.edu

Kabir Kang
College of Computing
Georgia Institute of Technology
kabirkang@gatech.edu

Nikhil Thomas
College of Computing
Georgia Institute of Technology
nikhil.jt@gatech.edu

Abstract

Accuracy losses due to distribution shifts between training and test data significantly impact the effectiveness of models deployed in new environments. Domain generalization is a critical challenge in machine learning, particularly given the high cost of acquiring labeled data. This paper proposes a novel architecture to improve domain generalization using a Mixture of Experts approach. We apply a domain router to assign wildlife images to expert models that specialize in subsets of the iWildCam dataset while maintaining the ability to generalize to unseen domains. We expect our approach to outperform traditional single-model baselines by routing data from unseen domains to the most relevant expert models. This research provides a robust solution to the domain generalization problem without requiring additional labeled data, enhancing model adaptability in diverse real-world settings.

1 Introduction

The past decade has shown the strength of machine learning (ML) models that perform well in various fields, as long as they are trained with sufficient data that is distributed identically to real-world applications. As people continue to use machine learning models as a solution to problems that do not allow for margin of error, such as vehicle automation and anomaly-detection in the medical setting, the need for models that generalize to out-of-distribution (OOD) data grows. Distribution shifts, which involve the distribution of training data differing from that of testing data, significantly degrade the performance of ML systems that are used in real-world settings. As demonstrated in previous research, generalization to new test sets that are intentionally made to be as identically distributed to the original test data distribution as possible can still produce significant drops in accuracy [6]. It is often impossible for developers of real-world applications to acquire training data that will always be identically distributed to every setting the application is deployed in. Because of this, a key goal of ML researchers is to bridge this gap by building robust models that can generalize between multiple domains.

Domain generalization (DG) is the field of study that involves using ML methods to preserve good model performance in domains that are different from the training domain. The goal of DG is for

models that are trained on one or more source domains to generalize well to unseen test domains. For example, in the field of image classification, a model that is trained on different styles of animal art should be able to generalize and classify photographs of those same animals. In recent years, there has been significant progress in domain generalization in fields such as computer vision and natural language processing. A variety of techniques, such as data augmentation, meta-learning, and adversarial training, have been developed to enhance the robustness of models in out-of-distribution scenarios. Additionally, leveraging unlabeled data in combination with labeled datasets has shown promise, as it allows for more effective use of available information when encountering new domains.

An important part of current progress in domain generalization is learning domain-invariant features of data. This allows ML models to infer based on the features that are most likely to produce accurate predictions in real-world deployments. Another key idea in DG is to reduce overfitting to source domains by augmenting data in training. Augmenting data has also shown promise when unlabeled data is more abundant than labeled data[9]. Consistency training on a large amount of unlabeled data is an effective method to make models robust and invariant to input noise. By using advanced data augmentation methods on unlabeled data, it is possible to improve the diversity of training examples, which helps models generalize better to unseen domains. This process encourages the model to focus on core features of the data that remain consistent across variations, which reduces the reliance on domain-specific patterns and improves generalization.

Distribution shifts are often studied using data that are synthetically modified to be OOD. The shifts produced by these methods have limitations when the goal is to generalize models to real-world shifts. WILDS is a benchmark of 10 datasets that reflect natural distribution shifts. The dataset we will use from this benchmark is the iWildCam dataset, which contains wildlife images from over 3,000 camera traps that were set up around the world. The distribution shifts that we will take advantage of from this dataset come from the background, illumination, camera angle, and biodiversity present at each camera trap location. This dataset contains over 1 million images, 80% of which do not have labels for the animals photographed. This is ideal for experimenting with using semi-supervised learning as an approach to addressing distribution shifts.

1.1 The Intuition

Images that are taken at locations geographically near each other often share similar characteristics, such as the background/landscape, vegetation, and other environmental traits. It is also likely that close cameras share roughly equivalent distributions of animal labels. It follows that image data originating from the same geographical region can be described as "similar" from human perception.

Because of this, there may exist more nuanced features unique to an arbitrary region which could assist with the classification task. A traditional approach to this task—a single large model—might not capture such region-specific features. This raises a driving idea that a model trained exclusively on a region would yield better performance with respect to data from the region compared to a general model trained on the entire dataset. It could be argued that large models do this automatically through its hidden representations, but this is not an explicit guarantee, especially since modern vision models are known for their obscurity and "black-box" nature.

Extending this driving idea, our architecture leverages a Mixture of Experts (MoE) approach, where a domain mapper predicts a domain representation for each input image. The domain mapper and router dynamically assign inputs to specialized expert models. Each expert model is trained to specialize in a subset of the data, allowing the architecture to better capture nuanced, domain-specific features.

The router uses the output of the domain mapper to determine the most relevant expert for a given input. This dynamic routing mechanism ensures that each expert handles the subset of data it is best suited for. Even when evaluated against unseen domains, the router assigns inputs to the expert whose training domain representation most closely aligns with the predicted domain of the input.

This mechanism enables the MoE architecture to tackle the broader challenge of generalizability to unseen domains by explicitly leveraging the strengths of specialized experts and dynamically routing data based on domain representations.

1.2 Research Objectives

The primary goal of this project is to implement our domain MoE architecture that uses a router to assign each domain to a specific expert that each have their own specialization. We hypothesize that this architecture will display strong classification performance on unseen domains compared to a baseline of a single, large model. Additionally, we aim to compare our architecture both against and in conjunction with other domain generalization methods.

2 Related Works

Domain Generalization. In recent years, the domain generalization field has seen increasing development. Studies have shown that SOTA image classification models already struggle to generalize onto test sets from similar distributions because of the distribution gap, not to mention on OOD test sets [6]. Furthermore, pre-trained models are a specific modern use-case for domain generalization methods, as individual downstream tasks will likely have differing distributions.

Most domain generalization methods can be broadly placed into two categories: methods that attempt to make a fixed model architecture more resilient to distribution shifts by modifying the training regime, and methods that broaden the model architecture to allow it to handle more domains and better generalize to unseen domains [8]. This project is primarily interested in the latter group of domain generalization methods, but it is also important to note that the two types of methods are not mutually exclusive. It is possible to further combine multiple domain generalization methods to potentially see further performance gains on OOD test sets.

Domain Experts. One category of architecture-based domain generalization methods involves training an expert model for each domain. The intuition is that each expert will be better at distinguishing classes within their domain than a single general model could achieve, but this method is not without its limitations. The number of models increases with the number of domains, so when faced with memory or compute limitations, the individual models can end up being severely downscaled. Additionally, when faced with an unseen domain from the test set that does not directly fall under any of the expert models, there has to be a method to select which expert model(s) are relevant. In most cases, these methods rely on the assumption that the unseen domain is similar in a way to some source domain(s) seen during training, which may not always be the case.

One of the first domain generalization works with domain-specific experts uses a domain prediction branch in their architecture to use source domain experts to classify samples from potentially unseen target domains [5]. Specifically, the domain prediction branch takes in the input image and predicts a series of weights, which are then used to combine the outputs of the source domain experts.

Ensemble Learning and Mixture of Experts. Since the introduction of domain experts to this field, there have been multiple different approaches when it comes to combining the experts during both training and inference. One such approach utilizes ensemble learning, which simply combines the experts' outputs equally at inference time [10]. However, their framework additionally implements unsupervised domain adaptation using collaborative ensemble learning. This semi-supervised learning method uses the prediction of the most confident expert model to train the non-expert models on unlabeled data, similar to knowledge distillation.

A more modern approach utilizes sparse Mixture of Experts layers to weight the experts. Rather than treating the domain experts as separate models, this work modified the last 2 even layers by replacing the FFN with an FNN for each expert and combining their outputs using cosine routing [4]. Cosine routing scales the outputs by the dot product between the embedding of the input image and a learned embedding for each expert, which ideally represents the central embedding of that expert's corresponding domain.

3 Setting

Our experimentation makes use of the WILDS iWildCam dataset, derived from the original 2020 iWildCam competition [3, 1]. Each data example consists of a camera trap image, the label of the animal present in the image, and a uniquely identifying integer for the camera that the image is taken from, otherwise known as the domain. The problem setting is to devise a classifier that is able to correctly identify the animal label given an image that may originate from a domain completely

unknown to the model during its training (unseen). In other words, the classifier should be capable of generalizing to the task at hand, rather than its training data solely.

There are 182 different animal species labeled in this dataset, though not all domains have instances of every animal (for obvious reasons). The labeled portion of the WILDS iWildCam dataset contains 323 domains, amounting to 203,029 data examples, while the unlabeled portion dataset houses 3,215 domains and 819,120 examples [3, 7].

3.1 Dataset Challenges

The iWildCam dataset used in this project presents challenges that motivate specialized domain generalization approaches:

- **Label Imbalance:** The dataset contains a large number of examples but exhibits significant label imbalance. This imbalance stems from some domains having certain animal species that do not appear in other domains. Label imbalance can bias models towards certain labels that are over-represented, which impacts generalization. To address this, we developed an architecture that is intended to have specialized experts that can handle some domains better than others.
- **Unlabeled vs. Labeled Data Proportions:** The dataset includes a greater proportion of unlabeled examples from a large number of extra domains. Although we considered using Unsupervised Data Augmentation (UDA), our project focused on developing an MOE. We only used labeled data for this project. Future work could incorporate semi-supervised learning techniques to use the full dataset.

4 Methods

4.1 Architecture

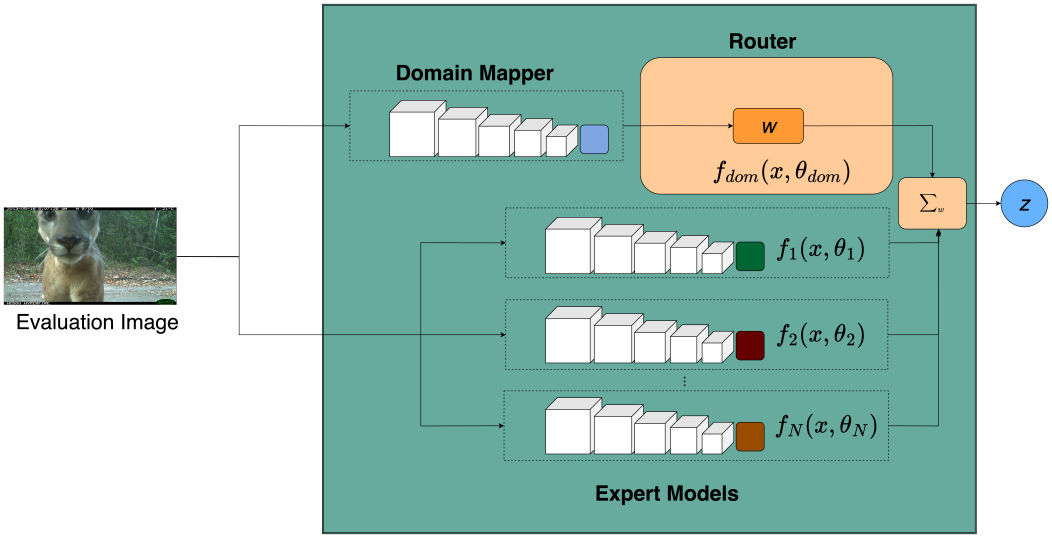

Figure 1: Our MoE Model Architecture. The input image flows through the domain mapper, router, and experts before producing a final prediction.

The Mixture of Experts architecture incorporates a router to assign input data to specific expert models based on their domain, as well as a domain mapper to adapt for OOD domains. This section details the functionality of the domain mapper and router along with how these components fit into the overall MoE architecture seen in Figure 1.

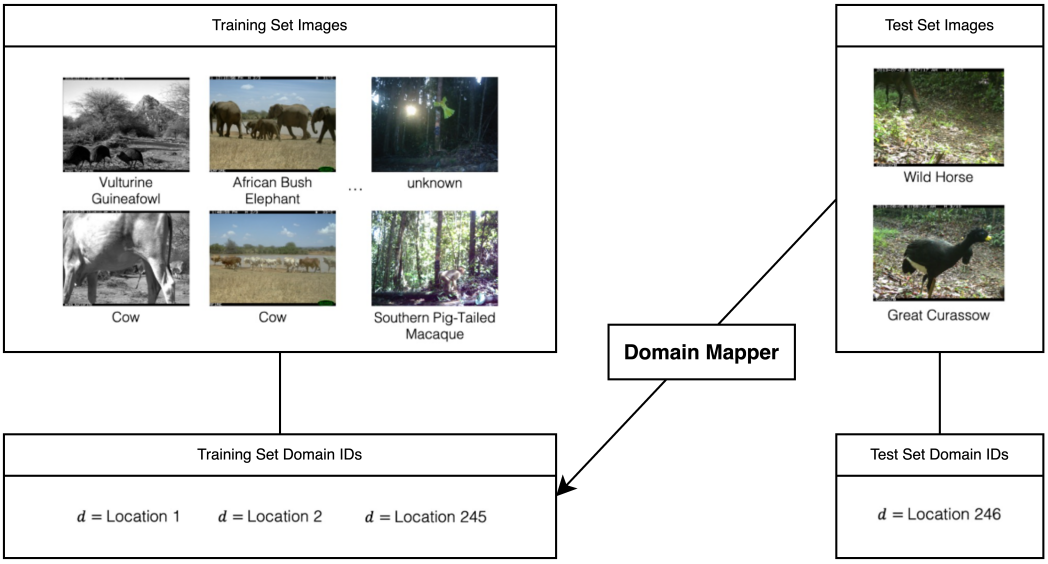

Figure 2: Domain Mapper

Figure 3: Router (W)

4.1.1 Domain Mapper

The domain mapper ($f_{dom}(x, \theta_{dom})$) is a key component of the system. It is responsible for predicting a representation for the domain d based on the input image x. This domain embedding is passed to the router to calculate routing weights. The mapper can optionally be trained alongside the MoE or kept frozen during training. The domain mapper is implemented as a pre-trained CNN (ResNet) fine-tuned on domain IDs rather than class labels.

4.1.2 Router (W)

The router maps the domain embedding to weights over the N_e experts. It is initialized with uniform weights, ensuring equal contributions from the experts at the beginning of the training phase. During training, the router learns to prioritize specific experts for each domain. At a predefined epoch, the router weights are converted into a one-hot representation, explicitly mapping each domain to a single expert.

4.2 Training Procedure

The training procedure for the Mixture of Experts follows the following sequence:

1. **Experts Initialization:** Experts are initialized using a snapshot ensemble [2], which is used to capture different local minima by cycling the learning rate. Another initialization method is tested where the experts are initialized using bagging aggregation where each expert is trained on a separate random sampling of the dataset (with replacement).

2. **Phase 1: Router Optimization:** The experts are kept frozen, while the router is optimized to assign weights for each expert based on the domain mapper's output.
3. **Phase 2: Router Snapping:** At a specified epoch, router weights are snapped to one-hot encodings, assigning each domain to a specific expert.
4. **Phase 3: Expert Fine-Tuning:** After snapping, the experts are unfrozen and trained further to specialize and diversify.

The held-out test split of the WILDS iWildData dataset is then used for evaluating both this architecture and a larger general model trained on the entire training dataset.

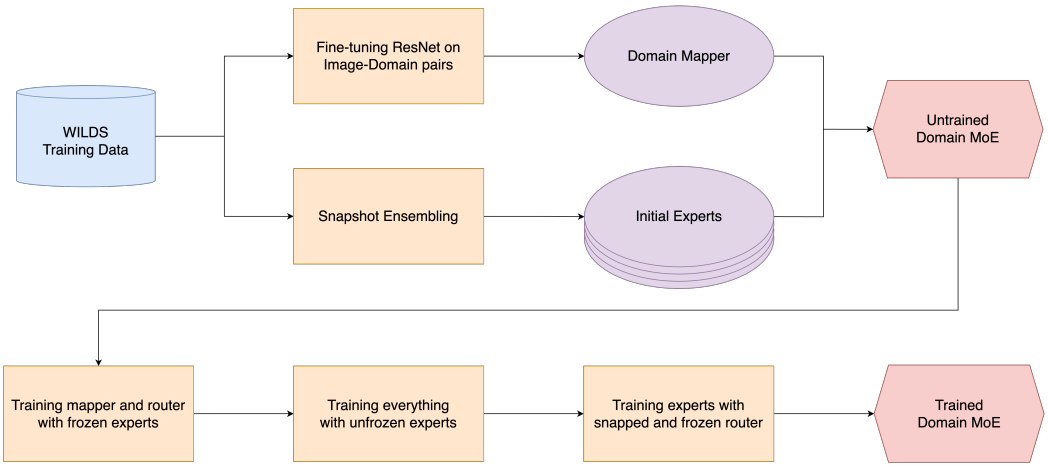

Figure 4: Training Procedure

5 Results

This section presents the performance of the models evaluated on the iWildCam dataset. The results are divided into in-domain and out-of-domain evaluations, highlighting the ability of the models to generalize across domains. The models tested include various ResNet architectures, Vision Transformers (ViT), and ensemble-based methods such as Snapshot Ensemble, Bagged Ensemble, and our proposed Mixture of Experts (MoE) architecture.

5.1 Out-of-Domain Performance

The out-of-domain results summarize how well each model generalizes to unseen domains. Figure 3 provides a comparison of accuracy, recall, and F1 scores across the models.

5.1.1 Observations

- **ResNet Baselines:** ResNet models performed well overall, with ResNet 50 achieving the highest accuracy among single models. This indicates that a deep pre-trained network remains a strong baseline for domain generalization tasks.
- **Snapshot Ensembles:** The Snapshot Ensemble with 5 experts slightly underperformed compared to ResNet baselines. The marginal improvement in recall for certain configurations suggests potential but insufficient diversity among experts.
- **Mixture of Experts (MoE):** While MoE models performed comparably to the Snapshot Ensembles, the anticipated improvement in generalization was not observed. Further analysis reveals issues such as limited expert diversity and potential overfitting to in-domain data.
- **Vision Transformers (ViT):** ViT models demonstrated competitive performance, particularly in F1 scores. However, their accuracy in out-of-domain settings lagged slightly behind ResNet 50.

Out-of-Domain (Test) Results

Model	Accuracy (%)	Recall	F1
ResNet 152	72.1133	0.272167	0.275299
ResNet 50	73.8193	0.260434	0.266368
ResNet 34	68.2059	0.223976	0.233204
Snapshot Ensemble			
SE 50, 5 experts	67.7526	0.255944	0.251981
SE 50, 3 experts	67.7315	0.262979	0.257199
SE 34, 5 experts	64.2495	0.227946	0.223529
SE 34, 3 experts	64.0672	0.226629	0.221488
Mixture of Experts (MoE)			
MoE 50, 5 experts	67.8951	0.248945	0.245326
MoE 50, 3 experts	68.1311	0.26692	0.260069
MoE 34, 5 experts	64.7613	0.22286	0.218446
Bagged Ensemble			
BE 50, 5 experts	68.9304	0.269803	0.25714
Vision Transformer (ViT)			
ViT 16	71.501	0.279906	0.283459
ViT 32	67.3296	0.187854	0.190388

Figure 5: Out of Domain Test Results

5.2 In-Domain Performance

In-domain results indicate how well the models perform on domains seen during training. Figure 4 presents the accuracy, recall, and F1 scores for each model.

5.2.1 Observations

- **ResNet Baselines:** ResNet 50 continued to perform well in in-domain settings, achieving one of the highest F1 scores across all models.
- **Snapshot Ensembles:** In-domain performance of Snapshot Ensembles was comparable to ResNet baselines, though the gains over single models were minimal, highlighting limited diversity among ensemble members.
- **Mixture of Experts (MoE):** MoE models performed similarly to Snapshot Ensembles, with no significant gains observed. The effectiveness of the router in assigning domains to appropriate experts requires further investigation.
- **Vision Transformers (ViT):** ViT models exhibited consistent performance across in-domain and out-of-domain settings, demonstrating robustness but no clear advantage over ResNet baselines.

5.3 Discussion of Results

The results highlight the following key findings:

- **Limited Expert Diversity:** Both Snapshot Ensembles and Mixture of Experts struggled to outperform ResNet baselines, which may be attributed to insufficient diversity among experts.
- **Router Performance:** The Mixture of Experts relied on a domain mapper and router for expert assignment, but the router's effectiveness in handling out-of-domain data remains a limitation.
- **Potential for Improvement:** Future work could explore more effective expert initialization strategies, dynamic router snapping schedules, and domain mapper improvements to enhance the generalization capabilities of MoE architectures.

In-Domain (Test) Results

Model	Accuracy (%)	Recall	F1
ResNet 152	72.6759	0.409623	0.416089
ResNet 50	72.5778	0.407456	0.41662
ResNet 34	68.028	0.352276	0.363949
Snapshot Ensemble			
SE 50, 5 experts	70.1005	0.409769	0.422219
SE 50, 3 experts	70.1128	0.410546	0.420224
SE 34, 5 experts	65.661	0.361691	0.363814
SE 34, 3 experts	65.8082	0.36083	0.364435
Mixture of Experts (MoE)			
MoE 50, 5 experts	69.9779	0.40949	0.417336
MoE 50, 3 experts	70.0147	0.40137	0.410823
MoE 34, 5 experts	64.3856	0.360808	0.366043
Bagged Ensemble			
BE 50, 5 experts	69.7694	0.402686	0.391315
Vision Transformer (ViT)			
ViT 16	69.0826	0.392117	0.388323
ViT 32	65.9431	0.348176	0.349186

Figure 6: In-Domain Test Results

Overall, ResNet 50 emerged as a strong baseline for both in-domain and out-of-domain settings, while ensemble-based methods such as Snapshot Ensembles and Mixture of Experts showed potential but require further refinement to achieve significant gains in generalization.

6 Discussion

Ultimately, we found our MoE architecture with snapshot ensembling to be less effective than we anticipated. Multiple reasons can be attributed to the MoE's underwhelming performance compared to more rudimentary architectures and training schemes. This section is dedicated to investigating and commenting on each of the potential causes. In addition, we discuss potential future directions to which this work can be expanded.

6.1 The Domain Router

One of the main components of this architecture is the Router W, which intuitively handles the assignment between the input data and the collection of experts within the MoE. One of the initial concerns with W is that it would immediately converge all domains to an arbitrary expert, which, by chance, performs slightly better than the rest. With the assumption that one expert generally does worse than an ensemble from which it derived from, it is plausible that a "lop-sided" router is the culprit.

However, uncovering the initialization of the router reveals that this is not necessarily the case. After training the MoE, we investigated the state of the snapped router. As shown in Figure 7, the responsibilities among experts were reasonable. Given 323 domains and 5 experts in the MoE, the "least responsible" expert is still assigned to 50 domains. In addition, we performed an ablation by randomly initializing W instead of setting all values to a constant. Still, the converged and snapped router, along with the MoE performance, were negligibly different, meaning that the router's initialization scheme is less significant than we anticipated.

```
(Pdb) torch.argmax(model.router.weight, dim=0)
tensor([0, 3, 1, 1, 4, 2, 0, 2, 1, 4, 2, 0, 1, 2, 4, 1, 0, 0, 0, 2, 2, 0, 1, 4,
        1, 2, 4, 2, 3, 1, 3, 2, 3, 4, 3, 1, 4, 1, 4, 1, 1, 2, 2, 1, 1, 3, 4, 4,
        0, 1, 1, 0, 0, 1, 3, 1, 1, 3, 1, 4, 4, 0, 1, 0, 0, 2, 0, 2, 3, 1, 2, 1,
        3, 0, 4, 4, 0, 0, 2, 4, 3, 4, 4, 1, 0, 0, 0, 2, 2, 1, 0, 1, 1, 2, 1, 1,
        1, 4, 3, 3, 0, 4, 2, 0, 2, 2, 0, 3, 2, 2, 0, 1, 0, 0, 4, 4, 0, 2, 4, 4,
        4, 1, 2, 4, 2, 3, 4, 4, 3, 1, 1, 2, 2, 1, 4, 1, 0, 1, 0, 2, 1, 3, 3, 4,
        2, 0, 3, 4, 1, 4, 0, 0, 1, 2, 4, 2, 0, 4, 3, 2, 3, 3, 4, 1, 3, 3, 0, 4,
        0, 1, 3, 1, 2, 0, 4, 1, 4, 3, 2, 3, 4, 3, 0, 1, 4, 3, 4, 3, 1, 1, 2, 1,
        2, 1, 0, 3, 0, 3, 4, 0, 2, 2, 3, 4, 1, 1, 1, 2, 3, 2, 4, 4, 3, 2, 3, 1,
        4, 1, 0, 4, 4, 0, 4, 2, 4, 0, 1, 1, 4, 0, 4, 4, 0, 1, 3, 1, 0, 0, 2, 2,
        2, 0, 3, 1, 1, 0, 1, 4, 2, 2, 1, 4, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 4, 4,
        1, 3, 3, 1, 2, 1, 3, 4, 2, 1, 3, 1, 1, 3, 4, 0, 4, 2, 1, 4, 3, 2, 3, 4,
        1, 2, 4, 1, 0, 2, 2, 3, 4, 4, 0, 4, 3, 3, 1, 0, 4, 1, 2, 4, 0, 0, 4, 3,
        3, 2, 2, 0, 2, 0, 4, 4, 0, 0, 0])
(Pdb) torch.unique(torch.argmax(model.router.weight, dim=0))
tensor([0, 1, 2, 3, 4])
(Pdb) torch.unique(torch.argmax(model.router.weight, dim=0), return_counts=True)
(tensor([0, 1, 2, 3, 4]), tensor([67, 77, 59, 50, 70]))
(Pdb)
```

Figure 7: Investigating domain-expert assignments made by the router. All experts are assigned a reasonable amount of domains.

```
tensor([[1.0000, 0.9490, 0.9416, 0.9539, 0.9513],
        [0.9490, 1.0000, 0.9510, 0.9497, 0.9492],
        [0.9416, 0.9510, 1.0000, 0.9435, 0.9432],
        [0.9539, 0.9497, 0.9435, 1.0000, 0.9460],
        [0.9513, 0.9492, 0.9432, 0.9460, 1.0000]])
```

(a) Pearson correlation matrix between experts in a snapshot ensemble.

```
tensor([[1.0000, 0.9442, 0.9368, 0.9487, 0.9355],
        [0.9442, 1.0000, 0.9509, 0.9517, 0.9235],
        [0.9368, 0.9509, 1.0000, 0.9471, 0.9178],
        [0.9487, 0.9517, 0.9471, 1.0000, 0.9247],
        [0.9355, 0.9235, 0.9178, 0.9247, 1.0000]])
```

(b) Pearson correlation matrix between experts in a MoE initialized with snapshot ensembling.

```
tensor([[1.0000, 0.6691, 0.6044, 0.6182, 0.5767],
        [0.6691, 1.0000, 0.6692, 0.6158, 0.6306],
        [0.6044, 0.6692, 1.0000, 0.6436, 0.6767],
        [0.6182, 0.6158, 0.6436, 1.0000, 0.5347],
        [0.5767, 0.6306, 0.6767, 0.5347, 1.0000]])
```

(c) Pearson correlation matrix between bagging experts.

```
tensor([[1.0000, 0.6686, 0.6089, 0.6159, 0.6660],
        [0.6686, 1.0000, 0.6721, 0.6085, 0.6669],
        [0.6089, 0.6721, 1.0000, 0.6391, 0.6983],
        [0.6159, 0.6085, 0.6391, 1.0000, 0.6164],
        [0.6660, 0.6669, 0.6983, 0.6164, 1.0000]])
```

(d) Pearson correlation matrix between experts in a MoE initialized with bagging experts.

Figure 8: Pearson correlation matrices

6.2 Limited Expert Diversity

When training the MoE architecture, it was noted that the training curves were stagnant. Assuming the router is working properly, an alternative scenario is that the experts initialized with snapshot ensembling may be too similar to each other. This likely deters the MoE from converging optimally, considering that, intuitively, the router expects diversity among the experts to in order to specialize each to a subset of domains.

To test this hypothesis, we performed a Pearson correlation test among the individual experts. This was done similarly in the snapshot ensemble study, where the paper found an approximate 0.88 - 0.93 correlation among each expert in their experimentation. However, our computed correlations were much greater, ranging from 0.94 - 0.95 (see Figure 8a). This confirms our suspicion regarding the lack of expert diversity, and may be the cause of stagnant training. Interestingly, from Figure 8b, the experts seem to diverge during the MoE training phase.

In addition to snapshot ensembling, we also experimented with bagging ensembling. Because this involves training each expert on an independent random sample of the original dataset, we believed that it would promote greater diversity between experts. This was proven true in Figure 8c, where the correlation between bagging experts ranges from 0.57 - 0.66, significantly lower than the correlation between snapshot ensembling experts. Our OOD results (Figure 5) find that the MoE initialized with bagging experts slightly outperforms the respective MoE initialized with the same number and model of snapshot ensembling experts, which seems to indicate that more diverse experts results in better performance. Furthermore, the effect of training the bagging experts in a MoE has the opposite effect compared to the snapshot ensembling experts—correlation scores actually increase across the board,

indicating the experts are converging together. This suggests that the ideal MoE performance occurs when the model correlation converges between the levels of the bagging and snapshot ensembling experts, so further investigation into the Pearson correlation scores may be warranted.

6.3 An Inherent Flaw with the Domain Mapper

Our domain mapper is a key element in our architecture that lets us deploy our solution to OOD data. As aforementioned, the domain mapper is trained entirely with labeled data, substituting the target label for the domain ID associated with the image. A significant "Achilles heel" with this method is the fact that we must assume the in-doman and OOD data belong in a distribution similar enough for the mapper to be effective. Clearly, this assumption is a strong one, and can explain the subpar MoE performance. Unfortunately, unless a portion of the OOD test data is labeled, it is not possible to determine the accuracy of the domain mapper. However, the mapper has a 89% accuracy on the in-domain test split, meaning that its accuracy on OOD test is likely to be worse.

6.4 Future Works

Despite the underwhelming performances seen in our experiments, we believe the proposed architecture has a lot of potential given more refinements and optimizations. During our experimentation, we have discovered branches of additional work that could advance our MoE concept, but are left for the sake of time. Below is a list of possible future directions of work:

- Rather than using a single domain ID as a feature, we could derive a feature embedding for a domain by taking an average over all images belonging to the domain. Then, we can augment W to map these higher-dimension features to experts.
- Currently, we unfreeze the experts and snap the router at specified epochs during the training stage. These could be changed as hyperparameters in the architecture, but more exploration (e.g. parameter sweep) can be done on these settings. Additionally, we could explore methods to automatically perform these actions after an accuracy threshold or some other heuristic.
- To verify the current domain mapper, we can leverage an image similarity algorithm to observe if the experts are being assigned to similar domains, or randomly hand-pick and investigate these assignments manually.

References

[1] Sara Beery, Elijah Cole, and Arvi Gjoka. The iwildcam 2020 competition dataset. *arXiv preprint arXiv:2004.10340*, 2020.

[2] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E. Hopcroft, and Kilian Q. Weinberger. Snapshot ensembles: Train 1, get M for free. *CoRR*, abs/1704.00109, 2017.

[3] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton A. Earnshaw, Imran S. Haque, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. Wilds: A benchmark of in-the-wild distribution shifts, 2021.

[4] Bo Li, Yifei Shen, Jingkang Yang, Yezhen Wang, Jiawei Ren, Tong Che, Jun Zhang, and Ziwei Liu. Sparse mixture-of-experts are domain generalizable learners, 2023.

[5] Massimiliano Mancini, Samuel Rota Bulò, Barbara Caputo, and Elisa Ricci. Best sources forward: domain generalization through source-specific nets, 2018.

[6] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet?, 2019.

[7] Shiori Sagawa, Pang Wei Koh, Tony Lee, Irena Gao, Sang Michael Xie, Kendrick Shen, Ananya Kumar, Weihua Hu, Michihiro Yasunaga, Henrik Marklund, Sara Beery, Etienne David, Ian Stavness, Wei Guo, Jure Leskovec, Kate Saenko, Tatsunori Hashimoto, Sergey Levine, Chelsea Finn, and Percy Liang. Extending the wilds benchmark for unsupervised adaptation, 2022.

[8] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip S. Yu. Generalizing to unseen domains: A survey on domain generalization, 2022.

[9] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. Unsupervised data augmentation for consistency training, 2020.

[10] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain adaptive ensemble learning. *IEEE Transactions on Image Processing*, 30:8008–8018, 2021.